

# Test de QuickReport 4 Professionnel

par

Date de publication : 20/07/2005

Dernière mise à jour : 01/09/2005

Présentation et test de la version 4.06 professionnelle de QuickReport pour Delphi 2005. Cet article est destiné à ceux qui ont déjà utilisé QuickReport et qui sont intéressés par les nouveautés de la version 4 ainsi que par son utilisation dans Delphi 2005. Si vous souhaitez poster des commentaires sur cet article allez [ici](#).

- I - Introduction
- II - Installation
  - II-A - Installation de QuickReport
  - II-B - Installation des composants TQRChart et TQRDBChart
- III - Nouveautés de la version 4 de QuickReport
  - III-A - TQRAbsTable : rendez les états indépendants des connexions
  - III-B - Nouvelle fenêtre de prévisualisation
  - III-C - Images de fond et transparence
  - III-D - Exportation des TQRRichText
  - III-E - Les filtres d'exportation
    - III-E-1 - Filtre PDF
    - III-E-2 - Filtre Excel
    - III-E-3 - Filtre WMF
    - III-E-4 - Filtre CSV
    - III-E-5 - Filtre HTML
    - III-E-6 - Filtre XML
  - III-F - Rapport composé exportable ( TQRComposteReport )
- IV - Migration de Quick Report 3 à Quick Report 4
- V - Ajouter l'exportation des TQRChart vers PDF/HTML
- VI - Conclusion des essais
- VII - Téléchargement et liens

## I - Introduction

QuickReport est un générateur d'état directement intégré à l'IDE Delphi. Il permet la conception des états en tant que fiche sans faire appel à un programme extérieur.

QuickReport fut pendant longtemps préinstallé avec Delphi. Avec Delphi 7 il n'est plus installé par défaut mais les paquets sont toujours présents et facilement installables. La version fournie avec Delphi 6 ou 7 est la version 3.0 Standard de QuickReport.

La version 4.06 professionnelle est disponible directement sur le site de l'éditeur ici : [Home of QuickReport](#). Cette version intègre beaucoup de nouveautés que je vais vous présenter au cours de cet article.

## II - Installation

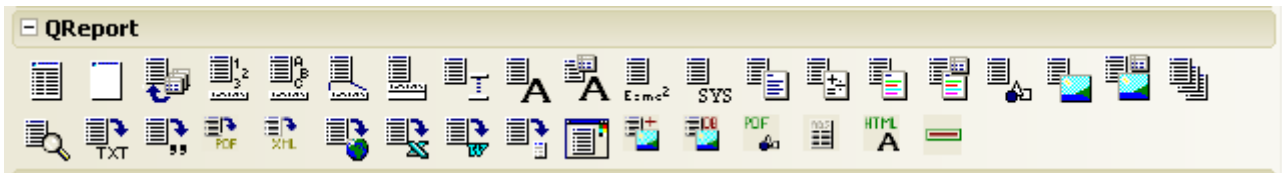
### II-A - Installation de QuickReport

QuickReport est fourni sous forme d'un fichier zip qu'il faut décompresser puis installer. Une fois installés les sources des paquets et des composants seront placés dans le dossier `C:\program files\QuickReportsX\`.

Pour installer le paquet dans Delphi, allez dans le menu *Composants* puis *installer des packages*.

Cliquez sur le bouton *Ajouter...* et sélectionnez le fichier `C:\program files\QuickReportsX\QR4DesignDX.bpl`.

Un nouvelle palette de composants nommée QReports sera alors disponible.



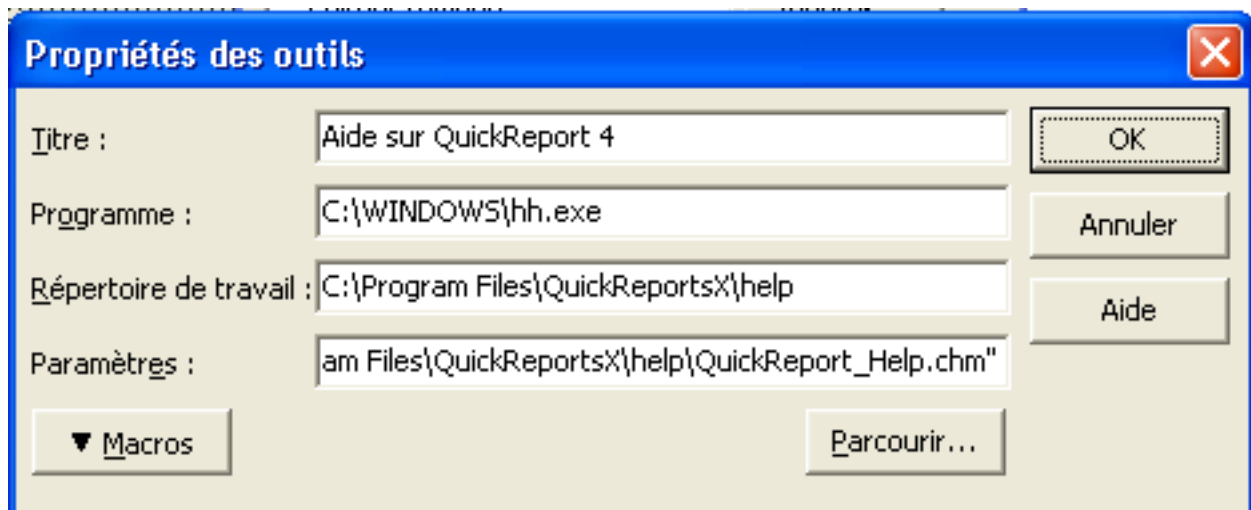
Pour tester l'installation il suffit de créer une nouvelle fiche, de placer un composant **TQuickRep**. Puis de placer un **TQRBand** et des QRxxx pour placer un texte et une image.

Dans la fiche principale il suffit d'appeler la méthode :

```
FenTest1.QuickRep1.PreviewModal;
```

QuickReport est livré avec un fichier d'aide au format CHM. Pour qu'il soit disponible facilement dans le menu outils de Delphi nous allons l'ajouter dans la liste.

Dans le menu *Outils*, sélectionnez *Configurer les outils...* et cliquez sur *Ajouter...* :



Le champ paramètres est "`C:\Program Files\QuickReportsX\help\QuickReport_Help.chm`"

### II-B - Installation des composants TQRChart et TQRDBChart

Ces deux composants ne proviennent pas de [QuSoft](#), mais de [Steema Software](#) concepteur de la suite de composants TeeChart.

Les sources des deux composants manquants sont disponibles sur cette page :

#### [TeeChart V4 files for QuickReport](#)

Téléchargez les sources données pour Quick Report 3 et Delphi 7 puis extraire les fichiers du ZIP dans le répertoire *C:\Program Files\QuickReportsX*.

Ouvrez le source du paquet d'exécution *C:\Program Files\QuickReportsX\TeeQR.dpk*.

Il faut modifier la liste des paquets requis en remplaçant QRpt par QR4RunDX :

```
requires
  vcl,
  vcldb,
  QR4RunDX,
  Tee,
  TeeDB,
  TeeUI;
```

Il faut aussi modifier la version des bibliothèques utilisées : (toujours dans le source du projet)

```
// Remplacez :
{$LIBSUFFIX '70'}
// par :
{$LIBSUFFIX '90'}
```

Vous pouvez maintenant compiler le paquet (aucune erreur ne doit apparaître normalement).

Ouvrez ensuite le source du paquet de conception *C:\Program Files\QuickReportsX\DclTQR.dpk*.

Supprimez la ligne **{\$R \*.res}** dans le source du projet sinon il ne sera pas possible de compiler le paquet à cause d'une ressource dupliquée.

Il faut de même changer le version des bibliothèques utilisées :

```
// Remplacez :
{$LIBSUFFIX '70'}
// par :
{$LIBSUFFIX '90'}
```

Vous pouvez maintenant construire et installer le paquet sans erreur.

Un nouveau composant doit alors être recensé : **TQRChart**.

### III - Nouveautés de la version 4 de QuickReport

Je vais présenter dans cette section les nouveautés principales de la version 4, telles qu'elles sont données ici : [Technical Information](#)

#### III-A - TQRAbsTable : rendez les états indépendants des connexions

Ce composant très simple d'utilisation est l'une des grandes nouveautés de cette version. Il permet de créer un ensemble de données virtuelles afin de simplifier tous les problèmes de connexion aux bases de données.

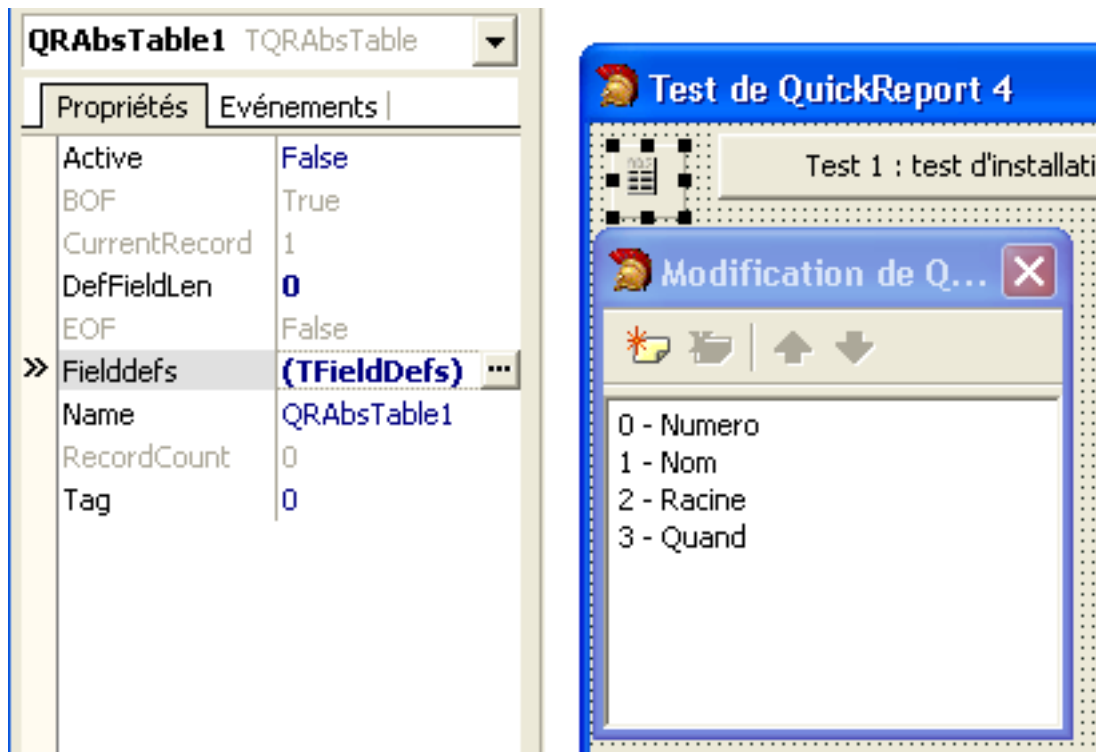
Ce composant comporte une liste de champs comme n'importe quel ensemble de données. Les valeurs de ces champs seront données par un événement de ce composant au fur et à mesure que l'état réclame des valeurs. Ce système simple permet d'utiliser un état standard et de gérer complètement par code les valeurs envoyées à l'état. Il devient donc très facile de créer un état à partir d'un fichier texte, d'un fichier XML ou directement d'une suite de calculs.

#### Mise en oeuvre

A titre d'exemple nous allons créer une table virtuelle contenant les champs suivants :

Nom	Type	Formule
Numero	ftInteger	recno
Nom	ftString (50)	recno en lettres
Racine	ftFloat	racine carrée de recno
Quand	ftDateTime	date dont recno est le numéro de jour

En utilisant le tableau ci-dessus on met à jour la propriété **FieldDefs** du composant.



Il faut ensuite donner le code de gestion de l'événement **OnSetFieldValue**.

- **Field** : Champ pour lequel la valeur est demandée
- **Buffer** : tampon servant à stocker la valeur du champ
- **MoreData** : Test de fin de table, il faut retourner *True* sur tous les enregistrements sauf le dernier où il faut retourner *False*.

En fonction du nom du champ passé en appel de la fonction, il faut retourner dans **Buffer** la valeur correspondante.

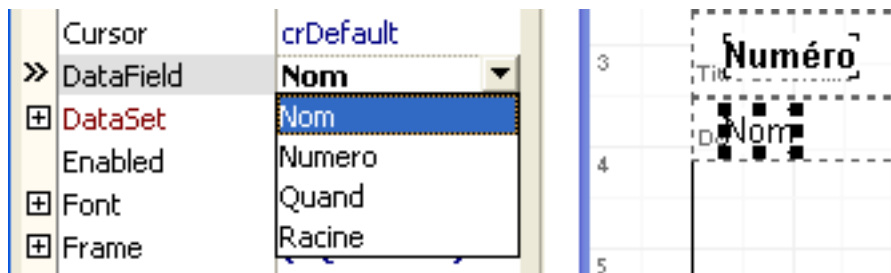
```

procedure TForm1.QRAbsTable1SetFieldValue(const DataStr: string; Field: TField;
  buffer: Pointer; var MoreData: Boolean);
var
  UnReel   : double;
  UnEntier : integer;
  UneDate  : TDateTime;
begin
  // Le premier champ est un entier
  If Field.FieldName='Numero' Then
  Begin
    UnEntier := QRAbsTable1.RecNo;
    CopyMemory( buffer, @UnEntier, SizeOf(UnEntier));
  End;
  // Le deuxième champ est une chaîne
  If Field.FieldName='Nom' Then
  Begin
    StrCopy(Buffer, Pchar(NombreEnLettres(QRAbsTable1.RecNo));
  End;
  // Le troisième champ est une valeur numérique
  If Field.FieldName='Racine' Then
  Begin
    UnReel := SQRT(QRAbsTable1.RecNo);
    CopyMemory( Buffer, @UnReel, sizeof(UnReel));
  End;
  // Le quatrième champ est une date
  If Field.FieldName='Quand' Then
  Begin
    UneDate := EncodeDate(2005,1,1)+QRAbsTable1.RecNo-1;
  End;
end;

```

```
CopyMemory( Buffer, @UneDate, sizeof(UneDate));  
End;  
// Test de fin de parcours de la table  
MoreData := QRAbsTable1.RecNo < 365;  
end;
```

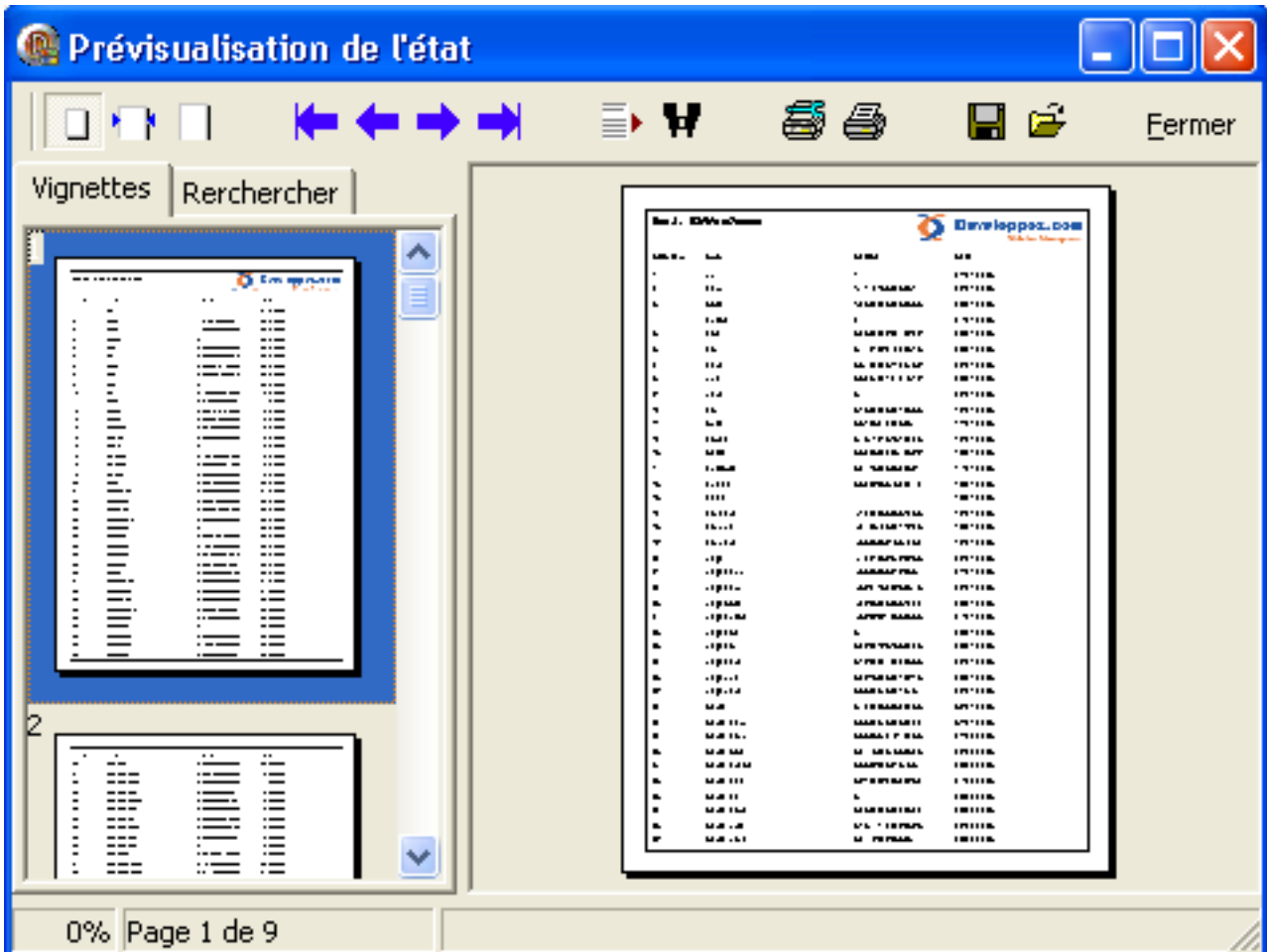
Une fois ceci défini, il faut mettre la propriété **Active** à *True* afin que la liste des champs soit disponible dans la conception des états.



A partir de cet ensemble de données, nous allons réaliser un état simple affichant simplement la table en entier. Pour ce faire il suffit de procéder comme n'importe quel état QuickReport et en utilisant **FenPrinc.QRAbsTable1** comme source de données.

### III-B - Nouvelle fenêtre de prévisualisation

La fenêtre de prévisualisation a été revue pour y ajouter deux fonctionnalités intéressantes.



L'onglet *Vignettes* permet d'accéder rapidement aux pages de l'état.

La deuxième fonctionnalité est la recherche d'un texte dans l'état. Il suffit d'utiliser le bouton rechercher puis de visualiser la liste des pages contenant l'occurrence dans l'onglet *Recherche*.

Il suffit ensuite de cliquer dans la liste pour visualiser la page concernée.

### III-C - Images de fond et transparence

Il est maintenant possible d'ajouter une image de fond sur un état et de gérer quels éléments seront transparents.

Pour mettre une image de fond, ajouter un composant **TQRImage** sur le **TQuickRep** (et non pas sur une bande).

Il faut donner au composant les dimensions et la position voulues sur la page, il n'est pas automatiquement ajusté à la taille de la page.

Dans le composant **QuickRep** il faut ensuite donner dans la propriété **BackImageControl** le nom du composant servant d'image de fond.

Un exemple de page générée avec une image de fond est donnée ici :

<http://nono40.developpez.com/tutoriel/delphi/quickreport/fichiers/essaiimages.html>

Notez que cette possibilité améliore beaucoup l'aspect visuel des documents.

Ceci fonctionne avec les états sur sortie graphique : imprimés, PDF, WMF et HTML.

### III-D - Exportation des TQRRichText

Les **TQRRichText** sont maintenant exportés dans les états graphiques. Ils sont intégrés en tant qu'image.

Dans les fichiers PDF les images sont incluses directement, dans les fichiers HTML les images sont incluses en fichiers séparés.

Le nom des fichiers images générés est le nom des composants TQRRichText allongé d'un indice incrémental afin qu'un RichText situé sur une bande puisse avoir une image différente sur chaque image de la bande.

Un exemple de page générée avec des **TQRRichText** :

<http://nono40.developpez.com/tutoriel/delphi/quickreport/fichiers/essaiimages.html>

### III-E - Les filtres d'exportation

La version 3.0 standard ne comportait que trois filtres : TXT, CSV et HTML. De nouveaux filtres sont désormais disponibles et le filtre HTML a été amélioré.

Pour les différents essais nous allons utiliser la table abstraite du premier test. Sur l'état un exemplaire de chaque filtre va être déposé afin de pouvoir exporter l'état sous ses différentes formes.

A noter qu'il suffit de poser un composant *filtre* sur un rapport quelconque pour que le filtre soit disponible pour tous les états de l'application.

#### III-E-1 - Filtre PDF

Ce filtre permet de créer des fichiers PDF en sortie d'état.

Pour l'utiliser, soit l'utilisateur enregistre l'état sous forme de PDF à partir de la fenêtre de prévisualisation, soit il utilise le code suivant pour générer le fichier directement :

```
Uses QRPDFilt
//...
procedure TForm1.btnPDFClick(Sender: TObject);
Var expFilt : TQRPDFDocumentFilter;
begin
  expFilt := TQRPDFDocumentFilter.create( 'D:\TEMP\QR\essai.pdf' );
  Try
    FenTest2.QuickRepl.ExportToFilter( expFilt );
  Finally
    expFilt.free;
  End;
end;
```

Je n'ai pas de remarques particulières sur l'utilisation de ce filtre. Un exemple d'état généré est disponible ici : <ftp://ftp-developpez.com/nono40/tutoriel/delphi/quickreport/fichiers/essai.pdf>

### III-E-2 - Filtre Excel

Ce filtre permet de créer des fichiers XLS en sortie d'état.

Notez que le format Nombre est bien géré en fonction des options régionales, ce qui constitue un plus pour les applications destinées à être diffusées dans plusieurs langues.

Le code pour créer le fichier XLS par code est le suivant :

```
Uses QRExport;
//...
procedure TForm1.btnXLSClick(Sender: TObject);
Var expFilt : TQRExcelFilter;
begin
  expFilt := TQRExcelFilter.create( 'D:\TEMP\QR\essai.xls' );
  Try
    FenTest2.QuickRepl.ExportToFilter( expFilt );
  Finally
    expFilt.free;
  End;
end;
```

### III-E-3 - Filtre WMF

Ce filtre permet de créer des fichiers WMF en sortie d'état.

Le format WMF ne supportant pas la notion de page, Quick Report va ici générer un fichier par page.

Le code d'exportation directe est différent des autres filtres, comme le format WMF est utilisé en interne par Quick Report on utilise les fichiers créés pour la prévisualisation :

```
procedure TForm1.btWMFClick(Sender: TObject);
Var expFilt : TQRWMFExportFilter;
begin
  expFilt := TQRWMFExportFilter.create( 'D:\TEMP\QR\essai.wmf' );
  Try
    With FenTest2.QuickRepl Do
      Begin
        Prepare;
        QrPrinter.ExportToFilter(expFilt);
        QrPrinter.Free;
        QrPrinter := nil;
      End;
  Finally
    expFilt.free;
  End;
end;
```

### III-E-4 - Filtre CSV

Ce filtre permet de créer des fichiers WMF en sortie d'état.

Ce filtre n'est pas nouveau, mais je vous donne ici des conseils quant à son utilisation.

La propriété **Separator** de ce filtre permet de choisir le caractère séparateur des champs.

Toutes les valeurs exportées par ce filtre sont entourées de " ", il est donc possible d'exporter des valeurs de type chaîne contenant le caractère d'exportation.

Les colonnes utilisées dans l'état sous forme de CSV seront définies en fonction des labels de la bande de titre.

Il est donc conseillé de tester l'exportation CSV afin de vérifier la disposition des colonnes en fonctions des Labels présents.

Le code pour générer un état CSV par code est le suivant :

```
procedure TForm1.btnCSVClick(Sender: TObject);
Var expFilt : TQRCommaSeparatedFilter;
begin
  expFilt := TQRCommaSeparatedFilter.create( 'D:\TEMP\QR\essai.csv' );
  Try
    FenTest3.QuickRepl.ExportToFilter( expFilt );
  Finally
    expFilt.free;
  End;
end;
```

### III-E-5 - Filtre HTML

Ce filtre permet de créer des fichiers HTML en sortie d'état.

Très pratique pour générer des pages web statiques à diffuser sur le web.

De nouvelles propriétés ont été ajoutées afin de faciliter la création et la composition du filtre.

- **ImageLinkPrefix** : Cette propriété va s'ajouter à tous les chemins d'accès aux images du document HTML final. Ceci permet de stocker les images dans un répertoire différent des pages html.
- **MultiPage** : Va générer le rapport sur plusieurs pages (un fichier html par page)
- **PageLinks** : A True cette propriété va ajouter des liens entre les différentes pages du rapport
- **PageLinkPrefix** : Cette propriété va s'ajouter à tous les liens entre les pages du document
- **PictureDir** : Chemin local de sauvegarde des images contenues dans l'état.

*La génération en mono-page d'un document de plusieurs pages ne fonctionne pas correctement.*

Exemple : <http://nono40.developpez.com/tutoriel/delphi/quickreport/fichiers/essaihtml1.htm>

#### Comment créer un jeu de pages.

La fonction multipage va créer autant de pages que l'état en comporte. Les pages vont être nommées du nom de fichier passé en paramètres plus le numéro de page. Par exemple l'état mono-page ci-dessus donne ceci en multi-pages : [Test multipage](#). Le nom de fichier passé en paramètres va être utilisé directement pour générer les liens. Si vous précisez un chemin en dur, ce chemin sera donné dans les liens html et seront donc invalides. Nous allons donc donner un chemin relatif en prenant soin de se positionner dans le bon dossier avant de lancer le filtre. Les pages seront alors créées au bon endroit et avec des liens relatifs facilitant la mise en ligne des pages. Notez que QuickReport permet en plus d'ajouter un préfixe sur tous les liens pour créer des liens web absolus.

Le code qui a généré le jeu de pages est le suivant :

```
procedure TForm1.btnHTMLClick(Sender: TObject);
Var expFilt : TQRHTMLDocumentFilter;
begin
  // On fixe le répertoire par défaut
  Chdir('D:\TEMP\QR\ESSAI\');
```

```
// On crée un composant et on génère les pages
expFilt := TQRHTMLDocumentFilter.create( 'essai.html' );
expFilt.MultiPage := True;
expFilt.PageLinks := True;
Try
  FenTest2.QuickRepl.ExportToFilter( expFilt );
Finally
  expFilt.free;
End;
end;
```

*La position des liens entre les pages n'est pas correcte en fonction de la longueur de la page.*

*Une des solutions est de mettre systématiquement une bande de pied de page non vide afin que le bas de la page soit toujours au même endroit. Les liens auront alors la bonne position.*

Avec cet outil vous pouvez générer des états en pages HTML facilement.

Exemple : <http://nono40.developpez.com/tutoriel/delphi/quickreport/fichiers/essai1.html>

### III-E-6 - Filtre XML

Ce filtre permet de créer des fichiers XML en sortie d'état.

Ce filtre permet d'obtenir une sortie sous forme de fichier XML de l'état. Il peut ainsi être utilisé comme base pour d'autres traitements ou directement affiché par un navigateur en utilisant une feuille de style XSL. La seule contrainte étant que le navigateur doit supporter les transformations XML/XSL.

La feuille de style est fournie par Quick-Report et est disponible dans *C:\Program Files\QuickReport\help\*.

Le composant **TQRXMLSFilter** ne fait qu'ajouter la possibilité de sauvegarder l'état sous forme XML à partir de la fenêtre de prévisualisation. Toutes les options du filtre ne sont pas accessibles par ce moyen, je vous conseille donc de générer les pages XML directement à partir du code :

```
procedure TForm1.btnXMLClick(Sender: TObject);
var xdoc : TQRXDocumentFilter;
begin
  xdoc := TQRXDocumentFilter.Create('D:\TEMP\QR\essai.xml');
  Try
    xdoc.Author := 'Nono40';
    xdoc.title := 'Test d'export XML/XSL';
    xdoc.Copyright := '(c) 2005 Bruno Guérangé';
    xdoc.XLStyleURL := 'essai.xsl';
    xdoc.XLEncoding := 'ISO-8859-1';
    xdoc.CompressImages := false;
    FenTest2.QuickRepl.exporttofilter( xdoc );
  Finally
    xdoc.free;
  End;
end;
```

Ceci permet notamment de gérer le codage du fichier, point important pour la facilité de lecture sur différents systèmes.

Un exemple d'état généré par le système est ici : <http://nono40.developpez.com/tutoriel/delphi/quickreport/fichiers/essai.xml>

*La feuille de style fournie avec Quick Report ne fonctionnera que sous Windows et seulement sous Internet Explorer. Ce qui limite sa diffusion.*

Vous pouvez télécharger la feuille de style corrigée ici :  
<http://nono40.developpez.com/tutoriel/delphi/quickreport/fichiers/essai.xml>

### III-F - Rapport composé exportable ( TQRComposteReport )

Les rapports composés peuvent maintenant être exportés à l'aide des filtres. Seuls les filtres XML, PDF et HTML sont supportés pour le moment.

Pour réaliser un rapport composé, il faut placer sur une fiche un composant **TQRCompositeReport** puis définir dans son événement **OnAddReports** la liste des rapports à ajouter.

Par exemple pour ajouter trois rapports séparés :

```
With QRCompositeReport1.Reports Do
Begin
  Add(FenRapport1.QuickRepl);
  Add(FenRapport2.QuickRepl);
  Add(FenRapport3.QuickRepl);
End;
```

Il est ensuite possible d'effectuer une prévisualisation avec la méthode **Preview**, une impression avec la méthode **Print** ou une exportation avec la méthode **ExportToFilter**.

Par exemple pour créer un fichier PDF :

<ftp://ftp-developpez.com/nono40/tutoriel/delphi/quickreport/fichiers/essaicomp.pdf>

```
procedure TForm1.btnCompPDFClick(Sender: TObject);
Var expFilt : TQRPDFDocumentFilter;
begin
  expFilt := TQRPDFDocumentFilter.create( 'D:\TEMP\QR\essaicomp.pdf' );
  Try
    QRCompositeReport1.ExportToFilter( expFilt );
  Finally
    expFilt.free;
  End;
end;
```

Et pour créer des pages HTML :

<http://nono40.developpez.com/tutoriel/delphi/quickreport/fichiers/essaicomp1.html>

```
procedure TForm1.btnCompHTMClick(Sender: TObject);
Var expFilt : TQRHTMLDocumentFilter;
begin
  Chdir('D:\TEMP\QR\ESSAI\');
  expFilt := TQRHTMLDocumentFilter.create( 'essaicomp.html' );
  expFilt.MultiPage := True;
  expFilt.PageLinks := True;
  expFilt.LinkFontSize := 12;
  Try
    QRCompositeReport1.ExportToFilter( expFilt );
  Finally
    expFilt.free;
  End;
end;
```

*Par défaut il n'y aura pas de saut de page entre les différents états du rapport composé. Pour ajouter un saut de page entre les états il faut appeler la méthode **NewPage** du **TQuickRep** dans l'événement **BeforePrint***

*d'une bande titre. S'il n'y a pas de bande titre, vous pouvez en ajouter une vide de hauteur nulle.*

## IV - Migration de Quick Report 3 à Quick Report 4

La migration de la version 3 à la version 4 s'effectue facilement. Des applications construites avec la version 3 standard de Quick Report livrée avec Delphi 6 compilent et s'exécutent sans changement sous Delphi 2005 avec Quick Report 4 professionnel.

Ceci est une bonne chose pour tous ceux qui envisagent de migrer leurs applications vers Delphi 2005 en conservant Quick Report.

Je n'ai donc rien de plus à signaler sur cette partie.

## V - Ajouter l'exportation des TQRChart vers PDF/HTML

Cette section n'est pas directement liée à Quick Report car elle traite de TQRChart. Mais afin de bénéficier des nouvelles exportations au format PDF, XML et HTML des composants **TQRChart** il faut modifier le source du composant.

Cette modification exporte automatiquement tous les **TQRChart** comme des images sans aucun ajout de code dans l'application finale.

Il suffit de modifier le code de **TQRChart.Print** :

```

procedure TQRChart.Print(OfsX, OfsY : Integer);
Var tmpMeta : TMetafile;
    QuickRect : TRect;
    tmpRect : TRect;
    tmpChart : TQRDBChart;
    tmpBitmap : TBitmap;

    RTFImage : TQRImage;
begin
    tmpChart:=Chart;
    if tmpChart<>nil then
    begin
        {$IFNDEF NOUSE_BDE}
        tmpChart.RefreshData;
        {$ENDIF}
        With ParentReport.QRPrinter do
        begin
            QuickRect:=Rect( Xpos(OfsX+Size.Left),
                            Ypos(OfsY+Size.Top),
                            Xpos(OfsX+Size.Left+Size.Width),
                            Ypos(OfsY+Size.Top+Size.Height));

            tmpRect:=tmpChart.GetRectangle;

            if Assigned( FONPrint ) then FONPrint( Self, QuickRect, tmpRect );

            Case FTeePrintMethod of
            qtmMetafile: begin
                tmpMeta:=tmpChart.TeeCreateMetafile(True, tmpRect);
                try
                    Canvas.StretchDraw(QuickRect,tmpMeta);
                finally
                    tmpMeta.Free;
                end;
            end;
            qtmBitmap: begin
                tmpBitmap:=tmpChart.TeeCreateBitmap(clWhite, tmpRect);
                try
                    Canvas.StretchDraw(QuickRect,tmpBitmap);
                finally
                    tmpBitmap.Free;
                end;
            end;
            end;
        end;

        if parentreport.Exporting then
        begin
            RTFImage := TQRImage.create(nil);
            tmpMeta := tmpChart.TeeCreateMetafile(True, tmpRect);
            Try
                RTFImage.Name := Self.Name;
                rtfimage.AutoSize := true;

                rtfimage.Picture.Assign( tmpMeta);

                // set up the image control export
                rtfimage.Size.Left := size.Left;
                rtfimage.Size.width := size.width;
                rtfimage.Size.top := size.top;
                rtfimage.Size.height := size.height;
            
```

```
rtfimage.Width := Round(size.width);
rtfimage.Height := Round(size.height);

TQRExportFilter(ParentReport.ExportFilter).acceptgraphic(
    qrprinter.XPos(OfsX + rtfimage.Size.Left),
    qrprinter.YPos(OfsY+ rtfimage.size.top ), rtfimage );

Finally
    rtfimage.free;
    tmpMeta.Free;
End;
end;
inherited Print(OfsX,OfsY);
end;
```

## VI - Conclusion des essais

La version 4.06 de QuickReport permet donc d'intégrer dans Delphi un générateur d'état simple et efficace. Comme il fonctionne directement au coeur de l'IDE il n'est nul besoin de fournir et d'installer de runtime avec vos applications.

Les nouveaux filtres d'exportation permettent de diffuser vos rapports sous différents formats simplement et rapidement.

Son intégration dans Delphi 2005 et la migration d'applications QuickReport 3 étant aisées, c'est un bon investissement pour faire évoluer vos applications.

## VII - Téléchargement et liens

Fichiers sources de cet article :

Miroir 1 : [Sources des exemples \[11Ko\]](#)

Dans le cas où le miroir 1 ne fonctionne pas :

Miroir 2 : [Sources des exemples \[11Ko\]](#)

Version PDF de cet article :

Miroir 1 : [Version PDF](#)

Dans le cas où le miroir 1 ne fonctionne pas :

Miroir 2 : [Version PDF](#)

[Site officiel de Quick Report](#)

[Site officiel de TeeChart](#)

[Page de téléchargement de TQRChart pour Quick Report](#)

Pour commander :

France : <http://www.sosdevelopers.com/>

Autres : <http://www.qusoft.com/>

Merci à [Olivier Lance](#) pour ses remarques et la correction orthographique.